REMARKS

Claims 1-23, 26 and 27 are currently pending in the subject application and are presently under consideration. Claims 1, 2, 8, 9, 11, 13, 21, 26 and 27 have been amended as shown on pp. 2-6 of the Reply. In addition, claim 7 has been cancelled and the limitation previously recited therein has been incorporated into the independent claims. Favorable reconsideration of the subject patent application is respectfully requested in view of the comments and amendments herein.

I. Rejection of Claims 1-18 and 27 Under 35 U.S.C. §102(e)

Claims 1-18 and 27 stand rejected under 35 U.S.C. §102(e) as being anticipated by Dandoy (U.S. Publication 2004/0230954). It is respectfully submitted that this rejection be withdrawn for at least the following reasons. Dandoy does not disclose, teach or suggest each and every limitation of the subject claims.

> A single prior art reference anticipates a patent claim only if it ***expressly or inherently describes each and every limitation set forth in the patent claim***. *Trintec Industries, Inc. v. Top-U.S.A. Corp.,* 295 F.3d 1292, 63 USPQ2d 1597 (Fed. Cir. 2002); *See Verdegaal Bros. v. Union Oil Co. of California,* 814 F.2d 628, 631, 2 USPQ2d 1051, 1053 (Fed. Cir. 1987). The ***identical invention must be shown in as complete detail as is contained in the ... claim***. *Richardson v. Suzuki Motor Co.,* 868 F.2d 1226, 9 USPQ2d 1913, 1920 (Fed. Cir. 1989) (emphasis added).

The claimed subject matter relates to an attributed debugging system that enables a developer to associate a specific view of an object for examining in a debugger. The specific view includes only object information necessary for the developer to debug the object. To this end, independent claim 1 recites *a computer-implemented attributed debugging system comprising a debugger that facilitates debugging of a computer software application, an expression evaluator associated with the debugger that determines values of one or more properties of an object of the computer software application being debugged,* ***the expressions evaluator determines the values of the one or more properties of the object based upon a display proxy implemented as a private nested class of the object, the display proxy provides relevant features of the object and conceals implementation specifics of the object and a***

variable display component that presents the determined values of the one or more properties of the object to a developer. Similarly, independent claim 27 recites *an expression evaluator inspects the at least one object to verify if each of the at least one object includes a display proxy defined as a private nested class of the object, the display proxy provides relevant properties of an associated object and conceals properties related to implementation of the object, the* ***expression evaluator examines the display proxy to determine debug information that includes values for the relevant properties of the at least one object***. Dandoy does not disclose, teach or suggest such aspects of the subject claims.

Dandoy relates to a user interface debugger. Dandoy discloses a debug agent that is combined with a software application. The debug agent collects execution data relating to graphical user interfaces during runtime. For instance, the debug agent correlates data objects (e.g., instantiated objects associated with a type class) to UI elements in the interface. The debug agent can further monitor the user interface for events. The debug agent provides execution data to a UI debugger or other debugger upon a user request. Moreover, the debug agent can alter properties of UI object during debugging. (*See* paragraphs 18, 21, 22, and 25).

In the Advisory Action dated July 30, 2008, the Examiner states that the debug agent disclosed in Dandoy is equivalent to an expression evaluator as recited in the subject claims. However, it is respectfully averred to the contrary. The debug agent does not determine one or more properties of an debugged object based upon a display proxy (claim 1) or examines a display proxy to determine debug information that includes values of relevant properties of the object (claim 27). Rather, the debug agent retrieves execution data. The execution data can include object properties, events associated with objects, run-time states of the application, etc. (*See* paragraph 18). In addition, the debug agent obtains execution data by reading state values stored in memory (e.g., states values of objects in memory) that are maintained by the application during execution (*See* paragraph 18). Accordingly, the debug agent in Dandoy does not employ, inspect and display proxy of an object as in the subject claims. Rather, the debug agent reads state values associated with objects themselves. Thus, Dandoy relates to inspecting objects while the subject claims relates to inspecting a display proxies associated with objects.

In the Advisory Action dated July 30, 2008, it is further contended that Dandoy discloses a display proxy that is implemented as a private nested class. However, the cited passages relate to the debug agent altering properties of an object associated with a graphical user interface element. For example, the cited passages disclose a user requesting to hide a window or other UI element (e.g., a button) within the user interface. Accordingly, the debug agent can toggle a property of the associated object to hidden to hide the UI element. However, Dandoy is silent regarding a display proxy. In Dandoy, the debug agent examines an object (e.g., a JButton object) and changes its properties (e.g., shown, hidden, colors, etc.). However, Dandoy fails to disclose examining a display proxy implemented as a private nested class (e.g., examining a JButtonDisplayProxy object defined as a nested class of JButton). A private nested class is an object type (e.g., a class) defined within a definition of another class (e.g., object type). Dandoy is silent regarding such aspects.

In view of at least the foregoing, it is readily apparent that Dandoy fails to disclose, teach or suggest all aspects of the claimed invention. Accordingly, it is respectfully requested that this rejection of independent claims 1 and 27 (and the claims that depend therefrom) should be withdrawn.

II. <u>Rejection of Claims 19 and 20 Under 35 U.S.C. §103(a)</u>

Claims 19 and 20 stands rejected under 35 U.S.C. §103(a) as being unpatentable over Dandoy in view of Bates *et al.* (U.S. Patent 6,961,924). It is respectfully requested that this rejection be withdrawn for at least the following reason. Claims 19 and 20 depend from independent claim 1 and Bates *et al.* does not rectify the deficiencies presented by Dandoy with respect to independent claim 1, as discussed above. Accordingly, withdrawal of this rejection is respectfully requested.

III. <u>Rejection of Claims 21-23 and 26 Under 35 U.S.C. §103(a)</u>

Claims 21-23 and 26 stands rejected under 35 U.S.C. §103(a) as being unpatentable over Bates *et al.* in view of Dandoy. It is respectfully requested that this rejection be withdrawn for at least the following reasons. Bates *et al.* and Dandoy, either alone or in combination, do not teach or suggest all limitations recited in the subject claims.

Independent claim 21 recites *a method facilitating attributed debugging comprising receiving a request to examine details of one or more properties of an object in a computer software application being debugged,* ***determining whether a display proxy attribute exists for the object, the display proxy is implemented as a private nested class of the object such that the***

display proxy within the definition of the object, the display proxy provides relevant properties regarding a state of the object and conceals properties related to implementation of the object, *creating a display proxy for the object in accordance with the display proxy attribute and examining the display proxy to determine debug information related to the object.* Similarly, independent claim 26 recites *means for determining whether a display proxy attribute exists in association with the object, the display proxy is implemented as a private nested class of the object such that the display proxy within the definition of the object, the display proxy provides relevant properties regarding a state of the object and conceals properties related to implementation of the object.* The cited references do not teach or suggest such aspects.

Rather, Bates *et al.* relates to debugging software code wherein the debugger includes additional descriptive material associated with variables beyond the values stored by the variables. In one embodiment, comments associated with a variable are displayed in a debugger in connection with the variable. The comments can be external (*e.g.*, stored in a database) or internal (*e.g.*, comments within source code). (*See* col. 3, ll. 39-51). In addition, information can be provided that describes the use of a variable. For example, attributes can be associated with variables that indicate the variable is a global, static, an index value, a parameter, a return value or a call value. (*See* col. 3, ln. 64 – col. 4, ln. 4). During the debugging process, the debugger checks to see if any attributes are associated with a variable and displays the attribute accordingly. (*See* col. 11, ll. 32-50). Thus, Bates *et al.* relates to providing information, such as comments or use information, in addition to the value of variables.

In the claimed subject matter, a display proxy, implemented as a private nested class of an object, is employed to generate debug information related to the object (*e.g.* a class variable or the like). The display proxy is examined by the debugger in place of the object. As recited in the subject claims, a display proxy provides relevant features of an object while concealing implementation specifics. For example, consider a HashTable object wherein keys are utilized in connection with values to store values in the HashTable. While debugging, a developer may only be interested in the keys and associated values of the HashTable. A debugger examining the actual object will reveal details that include complex variables specifying how the HashTable implements its functionality. A display proxy, however, will reveal the keys and values in a meaningful manner when examined in place of the object and conceal the complex variables relating to implementation. Bates *et al.* discloses examining variables themselves and not a proxy that only provides relevant features and not implementation specifics. For example, Fig. 7 of Bates *et al.* shows a variable "name" that includes both a value of a pointer as well as the value of the field pointed thereto. Thus, Bates *et al.* reveals the entirety of a variable. Further, Bates *et al.* is silent regarding examining a proxy that is a private nested class of an object to obtain variable value information.

Dandoy is relied upon to cure the aforementioned deficiencies of Bates *et al.* with respect to the claims. As discussed *supra*, Dandoy discloses a debug agent that is association with an entirety of an application and acts as a point of entry for execution. The debug agent monitors and/or collects data related to all user interface objects in the application. In the claimed subject matter, the display proxy is a private nested class of an object and, accordingly, has a one-to-one association. In addition, the display proxy is examined for information (e.g., object properties, etc.). In Dandoy, the objects themselves are examined to determine execution information. Thus, Dandoy fails to make up for the deficiencies of Bates *et al.* with respect to the subject claims.

In view of at least the foregoing, it is readily apparent that Bates *et al.* and Dandoy, alone or in combination, fail to teach or suggest the claimed invention as recited in independent claims 21 and 26 (and associated claims that depend therefrom). Accordingly, this rejection should be withdrawn and the claims allowed.

CONCLUSION

The present application is believed to be in condition for allowance in view of the above comments and amendments. A prompt action to such end is earnestly solicited.

In the event any fees are due in connection with this document, the Commissioner is authorized to charge those fees to Deposit Account No. 50-1063 [MSFTP579US].

Should the Examiner believe a telephone interview would be helpful to expedite favorable prosecution, the Examiner is invited to contact applicants' undersigned representative at the telephone number below.

Respectfully submitted,
AMIN, TUROCY & CALVIN, LLP

/Evan T. Perry/
Evan T. Perry
Reg. No. 62,190

AMIN, TUROCY & CALVIN, LLP
24TH Floor, National City Center
1900 E. 9TH Street
Cleveland, Ohio 44114
Telephone (216) 696-8730
Facsimile (216) 696-8731